SCORM and the Learning Grid

Fred Neumann, Rüdiger Geys

SCORM and the Learning Grid

Fred Neumann, Rüdiger Geys
FIM-NeuesLernen
Friedrich-Alexander-Universität Erlangen-Nürnberg
Konrad-Zuse-Straße 3 - 91052 Erlangen, Germany

{fred.neumann, ruediger.geys} @fim.uni-erlangen.de

The Sharable Content Object Reference Model (SCORM) is a profile of several elearning specification to ensure the reusability and interoperability of elearning content in web based Learning Management Systems (LMSs). Learning grids - learning environments built on the technology of grid services - are a promising new approach to enhance quality of elearning by overcoming the page oriented structure of the web. This paper investigates how SCORM can be used in conjunction with learning grids. After an introduction to the relevant aspects of SCORM two major scenarios are discussed: the use of grid enhanced content in a SCORM compliant LMS and the use of SCORM content in the context of a learning grid.

Keywords: eLearning Standards, ADL SCORM, Learning Grid

MOTIVATION

Compared with the high aims of a learning grid infrastructure, current web based learning management systems (LMSs) like WebCT or Blackboard are sometimes described as monolithic, server-centric and content delivery oriented. However, we should not underestimate the efforts and achievements of many institutions to formulate an elearning strategy, to select and establish an LMS, to customize and adapt it and to train their authors, tutors and lecturers in using it.

In the last years we saw a growing „common sense" on the functionalities and concepts of web based elearning. This is accompanied by the development of elearning standards and their increased acceptance by the vendors of authoring tools and learning management systems. Currently the specifications with highest impact are:

- IEEE LOM for metadata describing learning objects (LOs), enabling the search for content
- IMS Content Packaging to exchange LOs between authoring tools and learning environments
- AICC for the run-time communication between the content and the learning environment
- SCORM as a reference model to integrate the specifications above

For a certain period it seemed that especially the content packaging and run time environment are more adopted in commercial elearning applications for the use of commodity content modules available on the market. But meanwhile these specifications became also relevant in the academic scene, often after years of experiments with internally developed applications and contents using proprietary formats.

As a consequence, tools with scientific origin are modified to support elearning standards. For example ILIAS, an open source LMS popular at German universities, is now being extended to handle SCORM compliant content. This is a remarkable step, because light-weight, open source solutions are especially important for smaller, non profit organisation who try to apply elearning for new target groups and cannot afford expensive commercial servers that support the current elearning standards.

Grid computing is a promising new technological paradigm but the concepts for a learning grid are at the very beginning. To get a further understanding of their potential they may be developed in an idealistic way and evaluated in dedicated test beds. But in the mean time the development of conventional web based elearning solutions and standards will continue and the investments will increase. Therefore it is essential for the success and acceptance of a learning grid development to provide a smooth migration path from current web based to grid based solutions. This can be accomplished by taking into account the current exchange and interoperability standards in the elearning domain to enable:

- the use of grid services in a wide range of web based LMSs,
- the use of existing standard compliant content in a learning grid.

In this paper we will look at SCORM [1] for that purpose, as it provides an integrated view on a set of standards and specifications that focus on single aspects.

SCORM OVERVIEW

In 1997 the US Department of Defense (DoD) launched the Advanced Distributed Learning (ADL) Initiative as a collaboration between government, industry and academia to specify a learning environment that permits the interoperability of learning tools and course content. ADLs major outcome is the Sharable Content Object Reference Model (SCORM), a "collection of specifications adapted from multiple sources to provide a comprehensive suite of e-learning capabilities that enable interoperability, accessibility and reusability of Web-based learning content." (www.adlnet.org).

SCORM is an application profile for the use of existing standards and not a specification or standard in itself. The underlying specifications are slightly amended to be suitable for the DoD's education and training needs. However, the interoperability promoted by ADL became attractive to the commercial e-learning industry in general. The following table may give a rough impression of the current situation. It is remarkable that more software vendors are certified or declare themselves as compliant to SCORM then to its major underlying specifications (counted 3/2004):

cetis standards-compliant products directory (www.cetis.ac.uk)	• 19 products supporting IMS Metadata • 18 products supporting IMS Content packaging • 16 products supporting AICC • 25 products supporting SCORM
ADL (www.adlnet.org)	• 40 products certified • 96 vendors listed as SCORM adopters
AICC (www.aicc.org)	• 16 vendors certified • 21 vendors self-checked
IMS (www.imsglobal.org)	• 40 products in product directory (with conformance information given by vendors)

In January 2004 ADL released SCORM 2004, formerly referred to as SCORM Version 1.3. The major change is the introduction of the IMS Simple Sequencing specification. Other changes reflect the evolution of some specifications to IEEE standards or drafts:

- IEEE Learning Object Metadata (LOM)
- IEEE Draft XML binding for LOM
- IEEE ECMAScript API Interface for Content to Runtime Services Communication
- IEEE Data Model For Content Object Communication
- IMS Content Packaging
- IMS Simple Sequencing

SCORM concentrates on the interfaces between content and an LMS and does not specify the internal structure or features of an LMS. It is concerned with the functionalities to deliver, track, report and manage learning content, learner progress and learner interactions. For this purpose it defines the basic infrastructure to use content objects but not the context in which they are used. It is often stated that SCORM is best used in web-based training for individual learners with a self-paced and self-directed approach. But SCORM claims to be pedagogically neutral and may be used in any scenario, with or without the additional use of collaborative tools.

The reference model is divided into three parts ("Books") to which we will have a further look in the next sections: the Content Aggregation Model (CAM), the Run-Time Environment (RTE) and the Sequencing and Navigation (SN) Book. SCORM defines both, requirements for an LMS and for the content. Especially in the RTE and the new SN model the minimum requirements are very much higher for an LMS to achieve than for the content.

CAM: Content Aggregation Model

The CAM defines how instructional components are described with metadata and how they are organized and packaged. A content package includes all needed physical files and an xml based file named `imsmanifest.xml` describing the content and its structure. The manifest is composed of the following sections:

- *Meta-data* describing the content package as a whole
- *Organisations* describing the content structure. Multiple organisations are possible, each defining a hierarchical tree structure, optional context-specific meta data and optional definitions of sequencing and navigation behaviour
- *Resources* defining the bundled learning resources
- optional *(sub)Manifest(s)* describing logically nested units of instruction

SCORM defines two types of resources: Assets and Sharable Content Objects (SCOs). An Asset is a learning content in its most basic form, e.g. a HTML page, a picture or a video. It does not communicate with the LMS and is intended to be *completed* as soon as it is started. In contrast, an SCO can be composed of Assets. It communicates during run-time with the LMS and must not provide direct links to other SCOs so that the possibility of repackaging is ensured.

RTE: Run-Time Environment

If we look for the applicability of SCORM in a learning grid then the most interesting part is the run-time environment which specifies how an SCO is launched and how it communicates with the LMS. This section introduces the main RTE characteristics that will be discussed for a combination with grid services later. Figure 1 shows the RTE architecture:
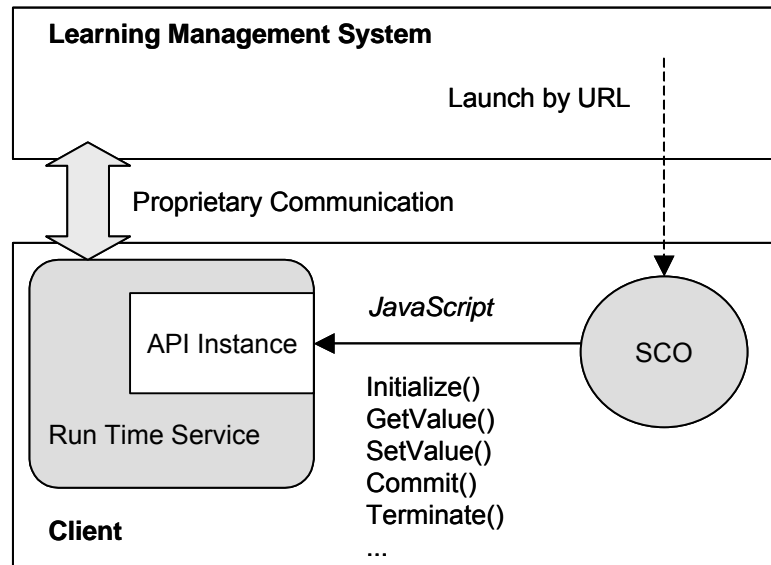

FIGURE 1: SCORM Run-Time Environment

The LMS uses the URL of an SCO defined in the manifest to launch the SCO by HTTP in the client's web browser, either in a child window or a child frame of the LMS window. Each SCO must be able to be launched this way. After that the SCO has to search in the DOM hierarchy of its window parents for an API object provided by a run-time service of the LMS that resides in the browser. SCORM does not specify how the run-time service is implemented and how it communicates with the LMS. All it defines are the API functions, how the API is found locally and that it must be called by JavaScript.

Every communication is initiated by the SCO. The SCO is responsible for starting the communication with an `Initialize()`and to end it with a `Terminate()` before it is removed from the browser. Between these calls it is able to read and write run-time data with `GetValue()` and `SetValue()`. Written data may be cached in the client by the run-time service. The SCO cannot check, whether data is cached but it may use `Commit()` to force the cached data being written back to the LMS. The remaining functions defined by the API are for error handling and diagnostic purposes.

The main interaction between SCO and LMS is done by exchange of run-time data according to a data model based on the IEEE draft standard with added requirements for the run-time behaviour. All data model elements are optional to an SCO but the LMS must provide them. Their initial values may be defined in the content package, by a learner registration process or by learner profiling requirements.

The data model elements have different purposes. Some are used to track status, scores and objectives of an SCO, some provide learner specific data and others have impact on the sequencing behaviour of SCOs. The run-time data is initialised at the beginning of a *learning attempt* which may span one or more learning sessions and can be suspended while keeping the actual values stored. Additionally a content package can define that the SCOs run-time data must be kept persistent over several attempts.

SN: Sequencing and Navigation

SCORM 2004 introduces the Sequencing and Navigation (SN) Book by applying the IMS Simple Sequencing specification to define how content objects may be sequenced through learner-initiated or system-initiated navigation events. For this purpose an organisation within a content package is derived into an *activity tree* for a learner. The leaf activities of that tree have content objects associated and the sequence of launched content objects builds an actual *learning experience.* The progression through learning activities may be sequential, non-sequential, user-directed or adaptive. It follows sequencing rules defined in the activity tree and takes the current RTE data into account.

IMS Simple Sequencing 1.0 is a rather complex specification and the amount of changes done by ADL in its pseudo code indicates that there is still some flow in the specification process. Therefore it will take reasonable time until the whole SCORM 2004 will be implemented by a wide range of systems and in the mean time SCORM 1.2 will remain a state of the art that supports only a hierarchical content menu and a simple forward and backward navigation through the SCOs.

SCORM AND GRID

Grid computing is a new technology for the flexible, secure and coordinated sharing of distributed resources and data. The Open Grid Services Architecture (OGSA) is built on web service technology and supports "transient services as named, managed entities with dynamic, managed lifetime" [3]. For elearning this allows the creation of learning grids – learning environments built on OGSA-compliant software and services. Learning grids promise to introduce a new quality of elearning by:

- enabling collaborative learning with peer-to-peer communication,
- making computing-intensive visualisations or simulations available for the learners,
- giving access to real-world data and resources like virtual laboratories.

Do SCORM and a learning have much in common? Looking at the core aims of SCORM (accessibility, adaptability, interoperability and reusability), they seem partially similar with the aims of grid technology. However, SCORM is focussed on the exchange and delivery of instructional content in web based LMSs, while grid technology is concerned with sharing of any resources in a distributed environments.
First we have to clarify what a learning grid will look like compared to a web based LMS. Will it be an infrastructure hidden behind a "portal" with conventional web access or will the learner's device be an active resource on the grid? Both architectures have their justification depending on the target group of learners. In the first case the grid functionality can be designed in any way, as long as the interaction with the user is done through HTML pages. But if a learning grid intends to overcome the page oriented design of the web then it must bring dynamic functionality to the learner's client. For best applicability this should be done in a transparent and system independent way, e.g. without the need for an extra installation, and requires an execution technology that is widely available in common browsers which currently means to use Java applets.

In [5] Pankratius and Vossen propose an elearning grid architecture that consists of two major parts: an LMS that is internally based on web services and a core grid middleware. They also introduce the concept of a grid-enabled learning object (GLOB) that includes conventional learning resources as well as an application layer for the use of grid services. Figure 2 is a simplified illustration of their concept without showing the detailed internal services. At the initial login to the LMS the fabric layer of the grid is delivered to the learners device as a Java applet. The login service of the LMS is able to authentify the user and his device against the grid and may also be trusted to create or delete grid users. The LMS will request a TGT (ticket-granting-ticket) from the grid middleware and deliver it to the client. If a GLOB is delivered to the learner's device it is able to use the fabric layer and the TGT initially provided by the LMS to further request session tickets and directly access the grid services.
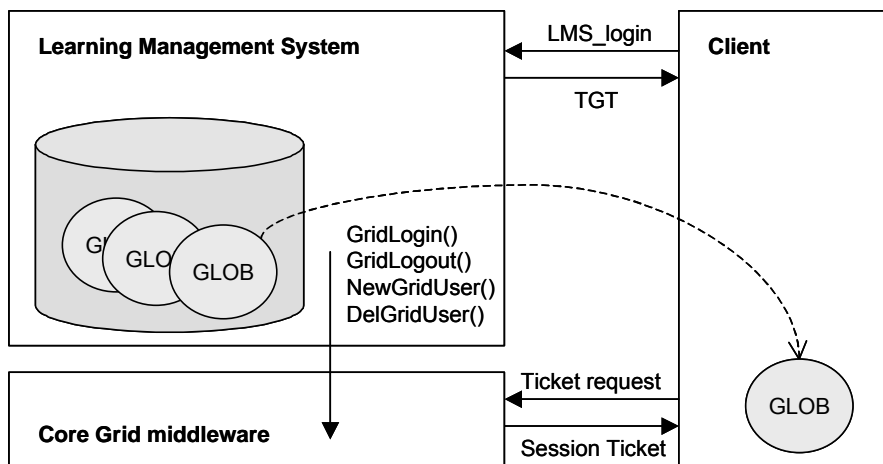

FIGURE 2: A proposed Learning Grid Architecture [5]

The proposed architecture describes a loose coupling between LMS and grid middleware with little requirements for the LMS: the call of a few operations from a grid login service and the delivery of a grid enabling applet to the client at login time. Therefore it seems possible to enhance nearly every web based LMS this way, regardless of whether the LMS is already internally built on web services or not. However, sometimes it will not be possible to introduce grid support by extending an LMS, for example in the following scenarios:

- An elearning content provider wants to offer grid services in conjunction with his content. To not restrict possible customers he must support as many platforms as possible.
- A lecture chair wants to offer grid based applications for a virtual course but has to use a campus-wide LMS that can't be modified.

There are many possible scenarios in which grid services can be integrated into a learning content: database searches, exercises, simulations, virtual laboratories or visualisations. For that we could ask whether it would be possible to realize GLOBs as grid enabled SCOs and use them in an LMS without the need to modify it.

Grid enabled SCORM content

In the following we assume that the whole functionality needed to access the grid is implemented by a Java applet ("grid applet") and included in a SCORM compliant content package. Depending on the learning purpose the applet may implement the whole fabric layer of the grid to include the learner's device as a resource on the grid or it may be just a client to access the grid via an ordinary web service. The most important aspect to clear is how to provide a single-sign-on from the LMS - that means how to authentify the LMS user for the desired service and how to transmit user related information between LMS and the grid.

For this purpose we have to take a look back to what the SCORM run-time-environment specifies. The grid applet must have access to the JavaScript API of SCORM for the exchange of data. The following data elements provided by the RTE are of special interest:

- `launch_data` is taken from the content package and is read-only
- `learner_id` is provided by the LMS and read-only
- `learner_name` is provided by the LMS and read-only
- `interactions` are used to store the results from interactive exercises
- `suspend_data` can be anything up to 4000 characters to be stored between sessions

Figure 3 shows a possible implementation. First the LMS launches the SCO containing the grid applet and the normal initialisation process is done. Now the applet has to authentify with the grid service. For this purpose it requests the `launch_data` from the LMS and extracts an included static key which is created by the vendor of the learning grid. The key is stored in the `imsmanifest.xml` of an SCO and may be set individually for an LMS or a course. Additionally the applet may determine its own origin with a `getCodeBase()` call. Due to JavaScript security restrictions in modern browsers the applet must come from the same domain as the API and so the code base helps to ensure that the applet is delivered from an LMS which is allowed to use the grid functionality [2]. The learner is identified by ID and the grid login is done with a combination of static key, code base and learner ID. The grid may log this data e.g. for billing purposes. A session ticket provided by the grid can be stored in the `suspend_data` element of the SCO's run-time-data.
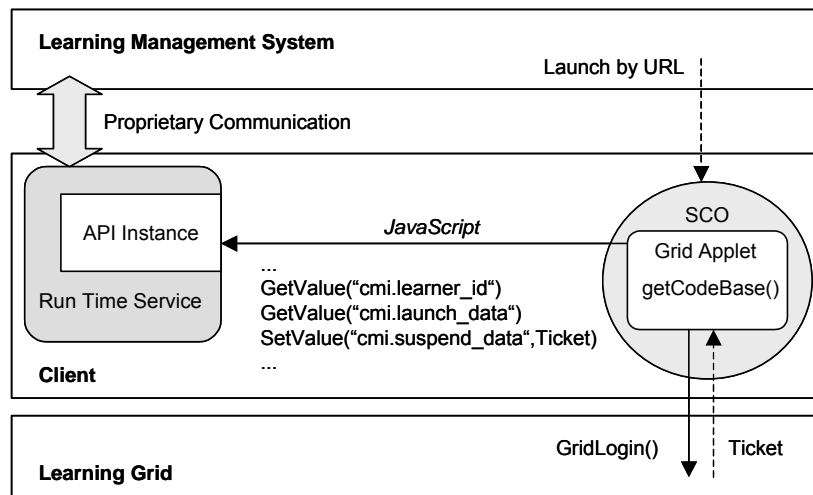

FIGURE 3: Using Grid enhanced SCOs

How secure is the proposed implementation? This questions is about preventing the grid from being misused (e.g. for stealing computing power) and preventing user accounts from being misused (e.g. for faking their activity). The communication between LMS and run-time-service is not specified and may have lower security than the grid provides, e.g. it may not be encrypted. Security is then determined by the weakest part - the LMS. Theoretically it is also possible to simulate an SCO session, extract the authentication from the LMS and to store and recompile the grid applet. Then a „cracked" standalone applet can be created for the use of a grid service with the ID of another user. However, this will mean a reasonable effort and will leave traces if both LMS and grid service log their accesses, so we can assume a medium security level. The problem comes from the fact that the authentication is based on static information (key and `learner_id`) delivered from the LMS to the client. Solutions to overcome this weakness may be applicable in special scenarios:

- The LMS is able to create session-tickets and deliver them to the SCO, e.g. in an extended run-time data model element. However, this would require the inclusion of a grid client on the LMS server and a modification of the LMS.
- In an Intranet the learners device may already be a resource on the grid. Then the grid applet in the SCO can use the local host and user for an authentication.
- The learner has to manually authentify for the grid at least once and receives a ticket that is stored in the suspend_data. Subsequent authentication may be necessary after time-out of the ticket.

A grid enhanced SCO may be designed to support all the scenarios with priority for the most advanced authentication method but a cascading fallback mechanism to the others.

Aside from security issues, a content oriented solution will not be appropriate for all purposes of grid enhanced elearning. A learner will work on content, but also use content-independent communication and collaboration facilities, e.g. for getting tutorial support or working in groups. A grid solutions for general supporting tools shouldn't be embedded in an SCO that is only accessible on a table of contents. SCORM can't be a help for this purpose and therefore the LMS has to be extended. An introduction of grid support may follow the architecture described in [5] but with the grid applets being directly embedded in the user interface of the LMS.

SCORM Content in Learning Grids
The obvious requirement for a SCORM compliant learning grid solution is to handle existing SCORM compliant content, at least to import it as content packages and to launch and track the SCOs for the learner. The question is how this content can be seamlessly integrated in a highly collaborative, grid-enabled environment. At present the reusability of SCOs prohibits a tight coupling of content with additional functionalities provided by an LMS like forums, conferences, or advanced tele-presence tools available in a grid infrastructure. The IMS Simple Sequencing provides minimal hooks for the invocation of so called "Auxiliary Resources", however SCORM does not yet define their further behaviour and ADL recommends content developers and LMS vendors to use them with extreme care to ensure future interoperability [1: SM-5-11]. The other direction is of equal interest: how may content objects easily be used by the learners as references in communication messages? SCORM requires that each SCO must be launched by its URL, but it may not be possible to simply copy it from the browser location into a message because it is hidden or includes user-specific parameters.

The conceptual relationship between content and collaboration is intensively addressed by the IMS Learning Design [4] which specifies an XML based language to describe complete learning scenarios including group work or collaborative learning. It formalizes the definition of actors and their roles in a learning scenario, its activities and the use of learning objects or services in an activity. IMS Learning Design is not included in SCORM and it is open whether it will be in the future. But a learning design may be integrated in a content package and SCOs may be used in a Learning Design. A learning grid designed to support both may enable pedagogically well defined collaborative learning with the use of widely available content. Learning Design is a conceptual model and does not define the technical implementation and the process of instantiating the activities. IMS provides a best practice guide with suggestions for developing a runtime environment but these are not normative. This gives freedom for the design of a service oriented learning grid architecture, especially in the decision which services should run locally on the learner's device or will be called remotely and whether the collaborative actions will behave in a peer-to-peer or server mediated manner.

On the other hand, if it is intended to include SCORM content in a learning grid, the architecture must take care of the requirements for a SCORM run-time environment. SCORM proposes that an LMS has a service based architecture, but it doesn't define anything of it, even not whether the services are real or just internal functionalities of a monolithic system. If we look at the run-time architecture in figure 1, a first step towards an integration of SCORM and grid technology would be to specify the proprietary communication between LMS and the run-time service as a web or grid service. The API functions defined by SCORM must be translated to service operations. An SCO should have the choice whether to use the JavaScript API or whether to directly communicate with the service, for example, if it is launched from a different content server. In this context the following statement from the SCORM specification is of special interest:

> "There are ongoing research and development efforts investigating alternative methods (e.g., as a Web Service) for LMS vendors to provide SCOs access to the API Instance. (...) Future versions of SCORM may introduce other communication protocols that support the fundamental requirements defined by the IEEE standard." [1: RTE-3-26]

The current SCORM API service may be one of a set of services for a content to LMS communication and we could imagine a suite of conformance levels with the SCORM API as the lowest common denominator and other services as extensions. Once the communication between LMS server and run-time service is specified, generic run-time services can be implemented as a bridge between service oriented LMSs and API oriented content. The JavaScript API may also wrap the extended services so that content developers can use them in a familiar way. This may be done by the introduction of new API functions or by the extensions to the run-time data model, e.g. with information on user groups and shared data for collaborative work.

To give a practical example, one extension could be the support for collaborative annotations. The editing and storage of annotations would be implemented by the LMS but their appearance must be rendered by content specific functions. The content must be adaptive to enable or disable this feature depending on what conformance the LMS provides and it must be able to invoke the editing function. We could also imagine that the annotations may be extended with presence indicators showing whether the originator is currently online (or logged in to the LMS) and allowing to start a synchronous communication. Again, this information is provided by the LMS, but rendered in the content, if available.

CONCLUSION AND DISCUSSIONS

Using grid services in a SCORM compliant LMS may be possible without modification as long as the grid functionality can be wrapped by learning object in a didactically reasonable way and as long as a medium security is acceptable. Otherwise the LMS has to be extended with grid functionality. Using SCORM in a learning grid may not just mean to "simple implement it" but will lead to extensions of the run-time-environment defined by SCORM if a real integration of content and communication facilities is intended.

We see that many aspects need a further investigation and many issues are still open, both in the SCORM and the grid world. However, the main open question is: how relevant will SCORM be in the future? Will it be the dominating model for content sharing at the time when a learning grid infrastructure is ready to use? Critic voices argue that SCORM is pedagogically poor while at the same time being too complex and too difficult to implement. How high is the benefit for a learning grid to have access to existing SCORM compliant content and how large will this pool be? Should a learning grid have its own reference model with a different set of supported specifications, e.g. with IMS Learning Design instead of Simple Sequencing? It would be a considerable effort to repeat the integration and profiling work already done by ADL.

A strategy may be not to rely on SCORM for the core architecture of a learning grid but to check for each concept how it harmonizes with SCORM content. In parallel, providing pluggable SCORM compliant, grid-enhanced content packages and creating small and smart learning grid applications with guidelines on how to integrate them in any LMS will raise the attention of the elearning world. This may create the "critical mass" for a broad acceptance of a learning grid architecture that will lead to its inclusion in the main standardisation process for elearning.

REFERENCES:

[1]   Advanced Distributed Learning (ADL). "Sharable Content Object Reference Model (SCORM®) 2004"
       Available at http://www.adlnet.org

[2]   Advanced Distributed Learning (ADL). "Cross-Domain Scripting Issue". Paper, October 2003
       Available at http://www.adlnet.org

[3]   Foster, I. C. Kesselman, J. Nick, S. Tuecke. "The Physiology of the Grid: Open Grid Services Architecture for Distributed Systems Integration". Proc. 4th Global Grid Forum (GGF4) Workshop 2002
       Available at http://www.globus.org/research/papers/ogsa.pdf

[4]   IMS Global Learning Consortium, Inc. "IMS Learning Design Version 1.0 Final Specification"
       Available at http://www.imsglobal.org/learningdesign

[5]   Pankratius, V., G. Vossen. "Towards E-Learning Grids: Using Grid Computing in Electronic Learning". Proc. IEEE Workshop on Knowledge Grid and Grid Intelligence, Halifax, Canada, Oct 2003, pp. 4-15